ORIGINAL ARTICLE

Journal Section

Job Shop Planning and Scheduling for Manufacturers with Manual Operations

Longzhi Yang[1]* | **Jie Li**[1] | **Fei Chao**[2] | **Phil Hackney**[1] | **Mark Flanagan**[3]

[1]Department of Computer and Information Sciences, Northumbria University, NE1 8ST, UK

[2]Cognitive Science Department, Xiamen University, Xiamen, China

[3]NHS Business Services Authority, Newcastle upon Tyne, NE15 8NY, UK

Correspondence
Longzhi Yang, Department of Computer and Information Sciences, Northumbria University, NE1 8ST, UK
Email: longzhi.yang@northumbria.ac.uk

Funding information
National Natural Science Foundation of China, No. 91746103

Conflict of interest
Conflicts of interest: none

Job shop scheduling systems are widely employed to optimise the efficiency of machine utilisation in the manufacturing industry, by searching the most cost-effective permutation of job operations based on the cost of each operation on each compatible machine and the relations between job operations. Such systems are paralysed when the cost of operations are not predictable led by the involvement of complex manual operations. This paper proposes a new genetic algorithm-based job shop scheduling system by integrating a fuzzy learning and inference sub-system in an effort to address this limitation. In particular, the fuzzy sub-system adaptively estimates the completion time and thus cost of each manual task under different conditions based on a knowledge base which is initialised by domain experts and then constantly updated based on its built-in learning ability and adaptability. The manufacturer of Point of Sale and Point of Purchase products is taken in this paper as an example case for both theoretical discussion and experimental study. The experimental results demonstrate the promising of the proposed system in improving the efficiency of manual manufacturing operations.

KEYWORDS
Job shop planning and scheduling, fuzzy learning and inference system, fuzzy rule interpolation, genetic algorithm, manual

operation scheduling

1 | INTRODUCTION

The manufacturing sector thrives on implementing efficiency initiatives through measuring and improving Key Performance Indicators (KPIs). Manufacturing heuristics that help senior management decision making are highly praised, particularly if they can deal with the inherent complexity of noisy shop floor data affecting planning scenarios. Manufacturing efficiency can be significantly improved by employing an intelligent job shop scheduling (JSS) system as shown by Operation Research (OR) and Artificial Intelligence (AI) solutions using combinatorial optimisations aiming to reduce the cost based on a given cost function, such as making span or economic cost (Çaliş and Bulkan (2015); Chaudhry and Khan (2016)).

The success of intelligent JSS systems relies on an accurate estimation of cost, such as time of each individual operation on each discrete functional processing step at each operation/work centre, when the making span is taken as the cost function. Despite the machine processing time of each operation being readily estimable, based on the nature of the job and the characteristics of the machine, it is difficult to accurately estimate the completion time of whole series of complex manual operations, which often disables the JSS systems. For instance, the multiple complex manual operations of the bespoke print industry produce Point of Sale (POS) and Point of Purchase (POP) products, with particular emphasis on the complexity of 3D display products, and hard to estimate in advance especially for large seasonal shop promotions. An individual manufacturing job may require tens of operations, utilising several different machines coupled with manual tasks in different operation centres, and many of these jobs may need to be processed in parallel at any single time. Parallel lines of manual operations (or “lanes") are indeed common and expedient where possible; these “lanes" need to be planned alongside the job estimation for individual atomic operations such as the ‘time to glue’, ‘time to cut a shape’, ‘time to consolidate’. The scheduler plans the lanes to conform to the physical space available and sensible groupings of manual tasks. The existing system fails once the manual processes take precedent.

The paralysed JSS fails the business in two ways and could lead to late jobs or additional ‘same day’ despatch costs incurred. The Sales Team is denied an accurate and realistic long-term schedule of operations and capacity management. Quotations are made available to customers simply based on the nature of the job rather than a reflection of demand-supply relationship. The danger being to overcommit which exceeds the actual capacity and leads to additional problems in the scheduling having to content with unnecessary Work-In-Progress (WIP). Secondly, the effective ‘management planning horizon’ is artificially shortened to a period as short as a week, instead of maintaining a strategic month or quarter perspective. This creates a cycle of “flood then famine” as a period of frantic hyper activity is replaced with insufficient work and quiet factory with a frustrated Sales Team. This common issue exists currently in most manufacturers where problems are seen in despatch but caused by planning, scheduling and estimating short comings. No despatch orientated solution exists as the answer lays in empowering the scheduler with stronger tools.

This paper proposes a complete manual job scheduling solution to address these limitations using a genetic algorithm (GA) and an adaptive fuzzy inference system by further developing the seminal work reported in Yang et al. (2017b). The fuzzy inference sub-system accurately estimates the completion time of a manual operation under different situations. Through lack of viable data on manual operations and thus the exclusion of data-drive rule base generation such as Chen et al. (2018), the rule base of the fuzzy inference sub-system is initialised by expertise knowledge. The rule-base is then developed dynamically and adaptively upon its deployment whilst performing inferences. By taking the accurate completion time estimation as inputs, the GA calculates the optimal scheduling by considering not only meeting all the job deadlines, but also minimising the overall cost of all jobs. To demonstrate a ‘proof of

principle', a series of experiments were performed on data that simulate real world loading scenarios common in the manufacturing environment. The experimental results demonstrate the power of the proposed system in improving overall manufacturing efficiency.

The rest of the paper is organised as follows. Section 2 briefs the theory underpinning of the proposed system. Section 3 overviews the proposed global scheduling system. Section 4 details the fuzzy inference and learning system in manual task completion time estimation. Section 5 presents the proposed manual job shop planning and scheduling system. Section 6 shows the effectiveness of the proposed approach through illustrative and simulated examples. Section 7 concludes the paper with future work highlighted.

2 | BACKGROUND

The two integral components of the proposed system, that is the JSS system and the experience-based fuzzy inference system are reviewed in this section.

2.1 | Job Shop Scheduling

JSS is a classical operation research problem or combinational optimisation problem, which has been extensively studied by researchers in the fields of Operation Research and Artificial Intelligence (Pinedo (2015); Johnson and Montgomery (1974)). Briefly, a JSS problem involves a finite set of jobs to be processed on a finite set of machines. Each job comprises a set of operations that must be performed on one machine within a certain set of capable machines with various constraints and costs, in a given job-dependent order. Therefore, JSS is essentially a machine scheduling problem where jobs represent a series of activities and machines represent resources and each machine can process at most one job at a time. A typical objective of this process is to minimise the total completion time required for all jobs (i.e., the make span) or the economic cost, although other key performance indicators (KPIs) may individually or jointly be used in the objective function.

As a classical research topic in the operation research community, common approaches developed by this community are linear programming and dynamic programming, whilst general search and optimization techniques developed by the community of artificial intelligence (AI) are utilised or adapted to JSS solutions, such as breadth-first search, Genetic algorithm (GA), Beam search (BS). A job shop scheduling problem is often represented and solved as a constraint satisfaction problem (usually termed as SAT or CSP), which specifies both the mathematical representation of the problem and the corresponding solving solutions (Ghédira (2013)). A large number of solutions have been reported in the literature for CSPs, which revolve around a series of artificial intelligence technological advances that have occurred over the last three decades Pinedo (2015). Most of the JSS problems are NP-hard, with a small number of exceptions, that is the computational requirements for obtaining an optimal solution grow exponentially as the size of the problem increases.

JSS solutions, no matter what form proposed by the operation research community or the artificial intelligence community, can be classified into three categories (Çaliş and Bulkan (2015); Chaudhry and Khan (2016); Miguel and Shen (2003); Gomes et al. (2005)): 1) the exact approaches based on hard search, 2) hard search with heuristics, and 3) inexact approaches based on soft search. The exact approaches or hard search algorithms can produce optimal solutions, but they are of exponentially computational time complexity. Typical hard search algorithms include branch and bound, integer linear programming and dynamic programming, amongst others. Various heuristics have been developed to speed up the hard search, with or without sacrificing the optimal solutions. By contrast, the inexact ap-

proaches or soft search algorithms do not guarantee optimal solutions, but it generates near optimal solutions. In fact, soft search sacrifices the absolute optimal solutions for a reasonable computational effort and thus time span. Soft search approaches are developed mainly based on the advances in soft computing. Common soft search approaches include genetic algorithms, simulated annealing, ant colony optimisation and fuzzy logic, amongst others. These approaches provide a full spectrum of compromise and balance between time and performance, which provides the users a great selection of options for problem solving.

Genetic algorithms (GAs) computationally simulate the evolutionary process from nature to explore the optimal solutions from a large searching domain by repeating three basic operations, selection, crossover and mutation. As an adaptive heuristic search algorithm for solving both constrained and unconstrained optimisation problems, GA has been successfully and wildly applied to solve various JSS problems, such as Ak and Koc (2012); Wang et al. (2009). In particular, GA utilises a number of individuals (or organisms) that specially implement biologically inspired computational structures, most commonly chromosomes, to compose a population (Dawkins (1982)). Each individual of the population usually contains either the different job's sequence number or allocated machines' number for each job, which represents a valid scheduling solution. Based on the current generation of population, the genetic operators, crossover and mutation, are employed to randomly generate the next generation of population, thus to form another valid scheduling solution. These operations are repeated until an optimal scheduling solution, which usually determined by minimising the cost such as the financial cost or the temporal cost of operations, is discovered.

2.2 | Fuzzy Inference and Interpolation

Fuzzy sets and systems represent and reason on vague information that arises due to the lack of sharp boundaries (Zadeh (1965)). The most widely applied fuzzy systems are fuzzy inference systems, such as the Mamdani inference (Mamdani (1977)) and the TSK inference (Takagi and Sugeno (1985)). In particular, these inference systems are able to represent non-linear and high dimensional decision making problems as fuzzy rule bases. Rule bases are either translated from expert knowledge, or extracted from data sets (Mamdani (1977); Tan et al. (2016)). Given an input, a fuzzy inference system produces a system output by referring to those rules in the rule base whose antecedents overlap with the given input. However, a fuzzy inference system will fail if the given input is not covered by any rule in the rule base. Fuzzy rule interpolation (FRI) was proposed to address this limitation (Kóczy and Hirota (1993); Huang and Shen (2008); Yang and Shen (2013); Li et al. (2017)).

FRI is essentially a fuzzy extension of piece-wise linear or polynomial interpolation or extrapolation (Yang et al. (2017b)). Accordingly, FRI enjoys the advantages of both fuzzy logic in terms of uncertainty management and piecewise linear or polynomial interpolation or extrapolation by means of knowledge generalisation. Indeed, except being used as a supplementary to conventional fuzzy inference systems to work with sparse rule bases led by limited knowledge, FRI can also be used for system simplification for complex fuzzy models by omitting those rules that can be approximated by their neighbours. Given an input which does not overlap with any rule antecedent, the two closest neighbouring rules can be identified based on a given fuzzy distance metric, which is the aggregation (usually weighted average) of the distances between the antecedent items and their counterparts in the observation (Huang and Shen (2008)). Then, an indeterminate rule is generated such that its antecedents are as 'close' (given a fuzzy distance metric) to the given input as possible. From this, a conclusion is derived by ensuring the shape distinguishability between the conclusion and the consequence of the intermediate rule is equal to that between the antecedents of the interpolated rule and intermediate rule.

The technical details of the above approach can be found in Huang and Shen (2008), which are omitted here due to space limitation. FRI approaches have been further developed from different perspectives. For instance, adap-

tive fuzzy interpolation was proposed to guarantee the interpolated results are consistent throughout the inference processes (Yang and Shen (2009, 2011); Cheng et al. (2016); Yang et al. (2017a)); rough-fuzzy rule interpolation was proposed for both representing the knowledge involving higher order uncertainty and facilitating rule interpolation with such knowledge (Chen et al. (2016)); dynamic version of fuzzy rule interpolation was proposed to add significant rules dynamically (Naik et al. (2017)); and experience-based fuzzy rule interpolation enables rule base involvement and revision whilst performing fuzzy inferences (Li et al. (2016)).

The experience-based fuzzy rule interpolation is particularly useful in this work for manual task completion time estimation thanks to its ability in adaptively generating and revising the rule base when only limited training data and/or expert knowledge is available. In particular, this system firstly initialises the rule base with a very limited number of rules representing the limited incomplete knowledge. Then, based on the existing rule's usage frequency information, historic performance information and distances between observations and existing rules, the two rules with the most important significances are selected for FRI rather than the two closest neighbouring rules as implemented in Huang and Shen (2008). Finally, the rule base is adaptively generated and revised, which is guided by the discrepancy between the interpolated result and the actual ground truth.

3 | OVERVIEW OF THE PROPOSED SYSTEM

This work takes a typical manual operation centre, i.e., the collate and pack area in the print industry, as the example for theoretical development and experimentation. In particular, a collate and pack area deals with all the collating, packing and other hand finishing processes. In the print industry, this area is usually treated as a single cost centre in manufacturing Management Information System (MIS), although practically multiple production lines (usually named lanes in the industry) are often planned and applied by the area manager manually in the collate and pack area for the performance of multiple jobs in parallel in pursuit of running efficiency; this discrepancy basically disables the MIS. The proposed system herein accurately estimates the manual task completion time and automates the lane planning and scheduling.

Note that this work focuses only on the collate and pack operations for simplicity, although the inclusion of tasks on machines is a scaled up version of this proposed system with more variables and constraints. The proposed system in this work is illustrated in Figure 1, which takes a set of manual jobs as inputs. The input jobs are annotated by the sales team which describes the nature of the job. Then the completion times of these jobs on different lane setups are estimated based on the job annotations, which are in turn fed forward to an optimisation algorithm, GA particularly in this work, for job planning and scheduling. After the planing/scheduling is implemented, the actual time used for each manual operation is compared with the estimated one, and the rule base is updated based on this feedback. The proposed system is able to learn whilst it performs and thus it will generally evolve and perform better along time. The two key components of the system, including the fuzzy inference sub-system and the GA subsystem, are detailed in the next two sections.

4 | MANUAL TASK COMPLETION TIME ESTIMATION

The completion time of a manual task is estimated by the experience-based fuzzy interpolation system as introduced in Section 2.2, which is demonstrated in Figure 2. The system mainly comprises of three parts: a rule base, an FRI subsystem (particularly a transformation-based approach (T-based) used in this work) and a rule base revision mechanism. In particular, the rule base is initialised based on limited expert knowledge, or more specifically based on the

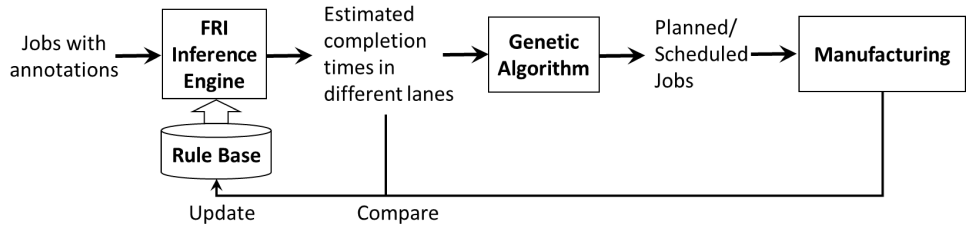


FIGURE 1 The overview of the proposed job planning and scheduling system

expertise of the manager of the collate and pack area in this paper. The rule base is then constantly revised by the rule base revision mechanism whilst the system performs inferences by the transformation-based FRI. These three main components are detailed in the following subsections.

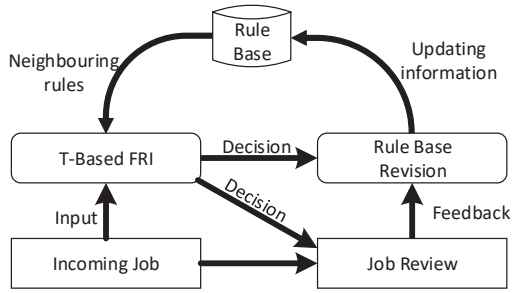


FIGURE 2 Experience-based fuzzy interpolation system

4.1 | Rule Base Initialisation

The lanes run in the collate and pack area are traditionally planned and scheduled by the area manager using their expertise knowledge. For simplicity and rule of thumb, the manager usually only implements three types of lanes: i) small lanes each implemented by one worker, ii) medium lanes each implemented by 4 workers, and iii) large lanes each implemented by 8 workers. Of course, other types of lanes may also be implemented for exceptional situations in order to achieve a deadline, but these are organised ad hoc without any serious planning and scheduling, and thus not considered in this work. The manager usually pursues both operational and managerial efficiency when setting up lanes. Note that in this work, it is assumed there is one shift per day, each lasting 8 hours. The expert knowledge is used in this work to initialise a fuzzy rule base for the estimation of the completion times of collate and pack tasks as discussed in the next subsection.

According to the expert knowledge, the time for completion regarding a given manual task on a particular lane

setup depends on a number of factors, typically including the object size, the task complexity, the number of parts, the total glue length, the number of applications of glue, the number of staples, and the number of folds; the feature values of each job are always provided by the sales team in the format of annotations. The unskilled nature of the assembly staff role implies a relatively short work life on this particular function. The worker may either move on to more skilled operations, be seasonal and only stay with the company in a matter of months, weeks, or even days, or threshold to an accepted level for their working life. The skill of the worker is therefore not a factor or parameter in the scheduling. This domain knowledge, which may vary from manufacturer to manufacturer, is used in this work to generate an initial rule base. The process of converting expert knowledge, usually expressed as linguistic rules, to fuzzy rules is beyond the scope of this paper, and a classical example of such process can be found in Mamdani (1977). Of course, if there are sufficient data available, data-driven approaches can also be used for rule base generation (Tan et al. (2016)), but this is usually not the case for most of the manufacturers due to the lack of data. As each piece of knowledge may be of different quality, a weight is also assigned to each rule indicating the quality of the rule. The variables used in the rule base and their domains are detailed in Table 1; and each rule is of one of the three following formats:

$$\begin{aligned} R_1^j &: \textbf{IF } x_1 \text{ is } A_1^j \text{ and } x_2 \text{ is } A_2^j \text{ and } \cdots \text{ and } x_7 \text{ is } A_7^j, \\ &\quad \textbf{THEN } z_1 \text{ is } B_1^j \ (w_1^j) \\ R_4^j &: \textbf{IF } x_1 \text{ is } A_1^j \text{ and } x_2 \text{ is } A_2^j \text{ and } \cdots \text{ and } x_7 \text{ is } A_7^j, \\ &\quad \textbf{THEN } z_4 \text{ is } B_4^j \ (w_4^j) \\ R_8^j &: \textbf{IF } x_1 \text{ is } A_1^j \text{ and } x_2 \text{ is } A_2^j \text{ and } \cdots \text{ and } x_7 \text{ is } A_7^j, \\ &\quad \textbf{THEN } z_8 \text{ is } B_8^j \ (w_8^j), \end{aligned} \tag{1}$$

where A and B are fuzzy sets; w represents the weight of the rule expressing the confidence of the corresponding domain knowledge; z_k represents the lane with k workers; and R_k^j represents the j^{th} rule regarding a lane with k workers.

TABLE 1 Variables and their domains

Variable	Meaning	Domain
x_1	Object size	{very small, small, medium, large, very large}
x_2	Task complexity	{very easy, easy, medium, complex, very complex}
x_3	No. of parts	A fuzzy integer number in a certin range
x_4	Total glue length	A fuzzy real number in a certain range
x_5	No. of glue applications	A fuzzy integer number in certain range
x_6	No. of staples	A fuzzy integer number in a certain range
x_7	No. of folds	A fuzzy integer number in a certain range
z_1	Time on a small lane	A fuzzy real number in a certain range
z_4	Time on a medium lane	A fuzzy real number in a certain range
z_8	Time on a large lane	A fuzzy real number in a certain range

Note that the rules above represent the rules of thumb in practice for manual lane planning and job allocations, but they usually do not lead to optimal operational efficiency. In order to enable the application of a JSS system for global optimisation of the manufacturing processes, an accurate estimation of the completion time for ad hoc lanes is of crucial importance. In other words, the times for completion need to be accurately estimated for all the combinations of inputs from the variable input domain with regard to various lanes of different sizes, which can be collectively represented as rules as:

$$\begin{aligned} R_k^j &: \textbf{IF } x_1 \text{ is } A_1^j \text{ and } x_2 \text{ is } A_2^j \text{ and } \cdots \text{ and } x_7 \text{ is } A_7^j \\ &\quad \textbf{THEN } z_k \text{ is } B_k^j\ (w_k^j), \end{aligned} \tag{2}$$

where $k = \{1, 2, \cdots, m\}$, m represents the largest number of workers on a single lane, and m takes 8 in this work. The complete rule base is impossible to be implemented based on the domain knowledge of the collate and pack area manager due to the complexity of the problem. Fortunately, this can be solved by adapting the recently proposed experience-based rule base generation and adaptation approach (Li et al. (2016)) as detailed in the next subsection.

4.2 | Fuzzy Rule Interpolation

The initialised rule base is very sparse. Suppose now there is a new collate and pack task which can be described as $x_1 = A_1^p, x_2 = A_2^p, \cdots x_7 = A_7^p$ as shown in the gray row in Table 2. This new input is not covered by any rule in the rule base. Then the neighbouring rules need to be identified to support the interpolation of completion times for the given new task based on different lane setups. Suppose that the closest neighbouring rules in the sparse rule base regarding the given input are identified and explicitly shown in Table 2, including:

$$\begin{aligned} R_1^f &: \textbf{IF } x_1 \text{ is } A_1^f \text{ and } x_2 \text{ is } A_2^f \text{ and } \cdots \text{ and } x_7 \text{ is } A_7^f \\ &\quad \textbf{THEN } z_1 = B_1^f\ (w_1^f), \\ R_1^g &: \textbf{IF } x_1 \text{ is } A_1^g \text{ and } x_2 \text{ is } A_2^g \text{ and } \cdots \text{ and } x_7 \text{ is } A_7^g \\ &\quad \textbf{THEN } z_1 = B_1^g\ (w_1^g), \\ R_4^h &: \textbf{IF } x_1 \text{ is } A_1^h \text{ and } x_2 \text{ is } A_2^h \text{ and } \cdots \text{ and } x_7 \text{ is } A_7^h \\ &\quad \textbf{THEN } z_4 = B_4^h\ (w_4^h), \\ R_4^l &: \textbf{IF } x_1 \text{ is } A_1^l \text{ and } x_2 \text{ is } A_2^l \text{ and } \cdots \text{ and } x_7 \text{ is } A_7^l \\ &\quad \textbf{THEN } z_4 = B_4^l\ (w_4^l). \end{aligned} \tag{3}$$

TABLE 2 The inference process

	x_1	x_2	x_3	x_4	x_5	x_6	x_7	k=1	k=2	k=3	k=4	...
	...	...	...	...	....	....	....	...	...	...	...	...
f	A_1^f	A_2^f	A_3^f	A_4^f	A_5^f	A_6^f	A_7^f	$B_1^f(w_1^f)$	x	x	x	...
g	A_1^g	A_2^g	A_3^g	A_4^g	A_5^g	A_6^g	A_7^g	$B_1^g(w_1^g)$	x	x	x	...
h	A_1^h	A_2^h	A_3^h	A_4^h	A_5^h	A_6^h	A_7^h		x	x	$B_4^h(w_4^h)$	...
p	A_1^p	A_2^p	A_3^p	A_4^p	A_5^p	A_6^p	A_7^p	?	?	?	?	...
l	A_1^l	A_2^l	A_3^l	A_4^l	A_5^l	A_6^l	A_7^l	x	x	x	$B_4^l(w_4^l)$	...
	...	...	...	...	....	....	....	...	...	...	...	...

In order to estimate the time for completion by a small lane with one worker, the two 'closest' rules R_1^f and R_1^g

in the rule base are identified based on a given fuzzy distance metric. From this, the value of variable z_1 for the given task can be estimated by means of fuzzy extrapolation using the approach briefed in Section 2.2. Similarly, the time for completion by a medium lane with four workers can be estimated using neighbouring rules R_4^h and R_4^l through fuzzy interpolation. From this, the time lengths of completion with 2-worker and 3-worker lanes for the give task can be interpolated from these two extrapolated and interpolated rules by artificially taking the number of workers on the lane as a rule antecedent attribute. The time lengths of completion based on any other lane setups can be either interpolated or extrapolated in the same way. Note that some of the interpolated/extrapolated rules that have been proven accurate which does not present in the existing rule base will be added in the rule base. This means that the rule base will become denser and denser after the deployment of the proposed system to enable the system to adaptively learn from practice.

The choosing of the closest neighbouring rules for interpolation/extrapolation can be different with the situation discussed above, and multiple options might be available. Suppose that the part of the rule base in the running example has been updated as illustrated in Table 3 (because interpolated/extrapolated rules R_3^p and R_4^p have been added in the rule base during the rule base revision which is discussed later in Section 4.3). Another new task presents which can be described as $x_1 = A_1^* \simeq A_1^p, x_2 = A_2^* \simeq A_2^p, \cdots x_7 = A_7^* \simeq A_7^p$ as shown in the gray row in Table 3. In this case, it is clear that neighbouring rules R_4^h and R_4^l can be used for interpolation and denote the interpolated results as $B_4^{p\prime}$. In the same time, by artificially taking the number of workers on lanes as an extra rule antecedent, the completion time can also be estimated from rules R_3^p and R_5^p and the result is denoted as $B_4^{p\prime\prime}$. In this case, the final result is a weighted aggregation of these two interpolated results in order to generate a global result, that is:

$$B_4^p = \lambda B_4^{p\prime} + (1-\lambda) B_4^{p\prime\prime}, \tag{4}$$

where λ is problem specific and 0.5 is used as a default value in this work.

TABLE 3 The evolved inference process

	x_1	x_2	x_3	x_4	x_5	x_6	x_7		k=3	k=4	k=5	...
	...	...	...	...	...	...	...	...	...	...	...	...
g	A_1^g	A_2^g	A_3^g	A_4^g	A_5^g	A_6^g	A_7^g	...	x	x	x	...
h	A_1^h	A_2^h	A_3^h	A_4^h	A_5^h	A_6^h	A_7^h	...	x	$B_4^h(w_4^h)$	x	...
$*$	A_1^*	A_2^*	A_3^*	A_4^*	A_5^*	A_6^*	A_7^*	...	$B_3^p(w_3^p)$		$B_5^p(w_5^p)$	...
q	A_1^l	A_2^l	A_3^l	A_4^l	A_5^l	A_6^l	A_7^l	...	x	$B_4^l(w_4^l)$	x	...
	...	...	...	...	....	....	....	...	...	...	...	...

4.3 | Rule Base Revision

The rule base keeps being revised based on its performance whilst it performs fuzzy inferences after the system is deployed. This is mainly achieved using a feedback mechanism. Upon the completion of a collate and pack task, the real completion time is compared with the estimated completion time; and the comparison result is used to determine the quality of the interpolated rule which is expressed as the weight of the interpolated rule. In particular, the weight w of an interpolated/extrapolated rule is defined as:

$$w = e^{-\frac{(t-t')^2}{a}}, \tag{5}$$

where $a(a > 0)$ is an adjustable parameter indicating the time error tolerance, and t and t' represent the actual and estimated completion times, respectively. It is clear that the maximum value of weight is 1, which represents the time predicted by the system exactly matches the time spent in the real-world case. In the work, $a = 30$ is applied, which is provided by the experts.

The interpolated rule may be used to replace an existing rule in the rule base, to extend the existing rule base or simply to be ignored. The framework of the rule base updating procedure is illustrated in Figure 3. An interpolated rule that has been proven accurate will be added into the rule base if there is not any similar rules included in the rule base. Suppose that R^* is an interpolated rule, which is represented as "**IF** x_1 is A_1^* and x_2 is A_2^* and $\cdots$ and x_7 is A_7^* **THEN** $z_k = B_k^*$ (w_k^*)", and that rule R is a rule in the existing rule base which is represented as "**IF** x_1 is A_1 and x_2 is A_2 and $\cdots$ and x_7 is A_7 **THEN** $z_k = B_k$ (w)". The similarity degree of the these two rules $S(R^*, R)$ can then be computed as:

$$S(R^*, R) = \frac{\sum_{j=1}^{7} S(A_j^*, A_j) + S(B^*, B)}{7 + 1}, \tag{6}$$

where $S(A_j^*, A_j)$ represents the similarity degree between the j^{th} antecedents of the interpolated rule R^* and the existing rule R, and $S(B^*, B)$ indicates the degree of similarity between the two consequences. Different approaches have been developed for similarity calculation between fuzzy sets (Chen and Chen (2003)). For instance, if triangular fuzzy sets are employed, each fuzzy set A can be represented as $A = \{a_1, a_2, a_3\}$, where (a_1, a_3) is the support of the fuzzy set and a_2 is the core or normal point of the fuzzy set. In this case, the degree of similarity between two triangular fuzzy sets A_j^* and A_j can be calculated as:

$$S(A_j^*, A_j) = 1 - \frac{\sum_{k=1}^{3} |a_{jk}^* - a_{jk}|}{3}. \tag{7}$$

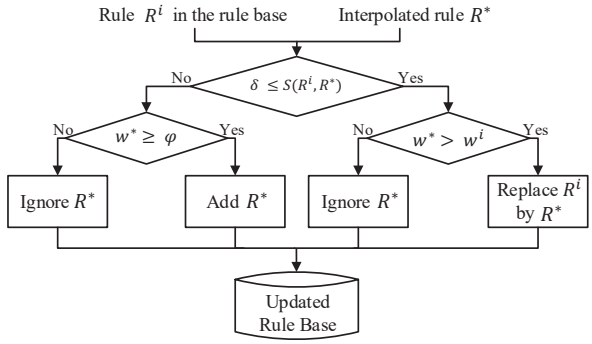


FIGURE 3 The flowchart of rule base revision

Given a similarity threshold δ, if the similarity degrees between the interpolated rule and all existing rules in the

rule base are less than δ, a potential new rule for the rule base is identified. From this, if the weight of this newly interpolated rule w^* is greater than a given weight threshold φ (i.e., $w^* > \varphi$), this newly interpolated rule will be added into rule base; otherwise, the interpolated rule will be ignored due to its poor quality. If the similarity degree between the interpolated rule and one or more existing rules are greater than the given threshold δ, the set of pair-wisely similar rules will be determined. In this case, the system will compare the weights of all these rules, and only keeps the rule with the highest weight value in the identified set. This action ensures that only the most accurate rule within the set is kept in the rule base, such that the rule base is concise and also of a good generalisation ability. By following the rule base revision progress, the number of rules in the rule base and accordingly the system complexity can be controlled by adjusting the similarity threshold δ and weight threshold φ.

5 | MANUAL JOB SHOP PLANNING AND SCHEDULING

Manufacturing efficiency can be significantly improved by employing the recent advances in artificial intelligence and this is true for the manufacturing of POS and POP which involves complex manual operations. A POS or POP is usually manufactured in multiple stages, such as printing, cutting, collating and packing, but, as stated in the Introduction section, only the manual collate and pack stage is considered in this paper. The objective of job planning and scheduling in this work is to minimize the make span, which not only leads to economic efficiency, but also provides informative information for the sales team such that they can take orders based on available manufacturing capacity. The task of an intelligent JSS system is then to find the best way of collating and packing with the shortest make span ensuring every job is completed within a given deadline, which requires the accurate estimation of completion time for manual tasks in different lane conditions based on the approach proposed in the last section.

5.1 | Problem Representation and Population Initialization

Genetic algorithm (GA), as an adaptive heuristic search approach for solving both constrained and unconstrained optimisation problems, has been successfully and wildly utilised to solve JSS problems. In particular, the algorithm firstly initialises the population with random individuals, and then selects a number of individuals for reproduction to produce the next generation of individuals by employing the genetic operators, typically crossover and mutation. The algorithm repeats this process until a satisfactory solution is generated or a pre-defined maximum number of iterations has been reached.

Assume that n incoming jobs are on the waiting list for manufacturing. A chromosome or individual, denoted as I, is designed to represent a potential 'solution' in this proposed system, as illustrated in Figure 4. An individual is formed by n genes, and each gene represents a job planning allocation. In particular, gene r represents that job J_r is allocated to lane k_r which is a lane with from 1 worker to 8 workers.

5.2 | Population Initialisation

The initial population $\mathbb{P} = \{I_1, I_2, \cdots, I_{|\mathbb{P}|}\}$ is formed by randomly assigning a job to an arbitrary valid lane. For a given job associated with a deadline, the required completion times for each of the 8 lanes with 1 to 8 workers can be accurately estimated by employing the proposed completion time estimation system as introduced in Section 4. A job can only be assigned to a valid lane on which the job can be completed within the required deadline. For instance, suppose that a collate and pack task requires to be done by the end of day 2 from the time of scheduling (i.e. two

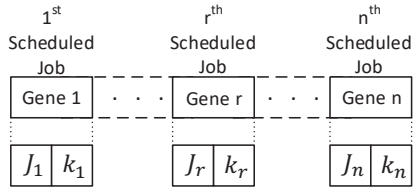


FIGURE 4 Chromosome encoding

shifts that is 16 hours), and that the estimated completion times on the 8 lanes from small to big are estimated as $z_1 = 18.3$ hours, $z_2 = 16.56$ hours, $z_3 = 14.53$ hours, $z_4 = 12.65$ hours, $z_5 = 10.74$ hours, $z_6 = 8.87$ hours, $z_7 = 6.95$ hours, and $z_8 = 5.1$ hours. It is clear based on the deadline that valid lanes are those with 3, 4, 5, 6, 7 or 8 workers. Small lanes with 1 or 2 workers are not valid as it will take over two days to complete the job on these lanes.

5.3 | Objective Function

An objective function is used in the GA to determine the quality of individuals. The aim of the proposed system is to optimise the lane planning and job scheduling, by minimising the time cost of the collate and pack operations and ensuring all jobs can be done within the specified deadline. Therefore, the objective function in this work is defined as:

$$min\ t, (t = t_1 + \cdots + t_n), subject\ to: \forall t_r, r \in [1, \cdots, n], t_r \leq d_r\ , \tag{8}$$

where t is the required time to complete all the jobs in the waiting list, t_r represents the estimated completion time for job J_r, and d_r denotes the given deadline for corresponding job r. The individual with the smallest value of t represents the optimal solution in the population.

5.4 | Selection

A number of individuals need to be selected to generate the next generation of population. The progress of the selection in this work is implemented by the fitness proportionate selection method, also known as the roulette wheel selection. Given the current population $\mathbb{P}$, if f_i is the fitness value of individual I_i in $\mathbb{P}$, its probability of being selected to generate the next generation is:

$$p(I_i) = \frac{f_i}{\sum_{j=1}^{|\mathbb{P}|} f_i}\ , \tag{9}$$

where $|\mathbb{P}|$ is the total number of individuals in the population, or population size. In the proposed system, the fitness value f_i of individual I_i is defined by adopting the linear-ranking algorithm (Baker (1985); Li et al. (2018)), which is defined as:

$$f_i = 2 - max + \frac{2(max - 1))(r_i - 1)}{|\mathbb{P}|}\ , \tag{10}$$

where max is a bias or selective pressure towards the fittest individuals in the population, r_i is the ranking position of I_i in $\mathbb{P}$.

5.5 | Reproduction

For a selected number of individuals, two genetic operators, crossover and mutation, are applied to breed the next generation of individuals, as shown in Figure 5. In this work, a two-point crossover operation is adopted, which exchanges the genes between a pair of parent individuates in certain order to generate a pair of offspring (Anand and Panneerselvam (2016)). In particular, the crossover operation firstly randomly selects two crossover points, say p_1 and p_2 in parents I_i and I_j, and then copies the genes between the two selected crossover points from I_i and I_j to the same positions of offspring I'_i and I'_j. After that, the rest fields of I'_i will be filled by I_j from the left side of the crossover point p_2 in a cyclic order. If any job in a gene already exists in I'_i, this gene will be skipped. The second offspring $I'_j$is generated by following the exact same operation. This crossover process is demonstrated in Figure 6, which takes 10 genes as an example.

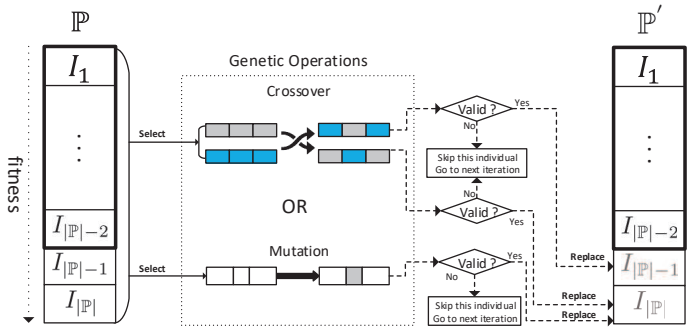


FIGURE 5 Procedure of reproduction

The second genetic operator is mutation, which is adopted in this work to maintain genetic diversity from one generation to the next. During the mutation operation, a gene from the parent is randomly selected and the planned lane in the gene is randomly mutated to another valid lane of a different size, thus to generate an offspring. The newly bred individuals and some of the best individuals in the current population $\mathbb{P}$ jointly form the next generation of the population ($|\mathbb{P}|$). Note that only one crossover or mutation operation will be allowed to occur per generation and the pre-defined rates are used to control the percentage of operations. In addition, in order to guarantee the generated offspring represent a valid solution, a deadline constraint is always applied to make sure each job can be done within the required deadline. If a produced individual satisfies the applied deadline constraint, which means all scheduled jobs can be finished before the deadline, this individual will be kept for the next generation of population. Of course, if the number of valid individuals in the old generation is less than $|\mathbb{P}|$ which is usually the case during the first stage of iterations, the produced individual is also kept (despite not being a valid solution, because it increases genetic diversity in the gene pool).

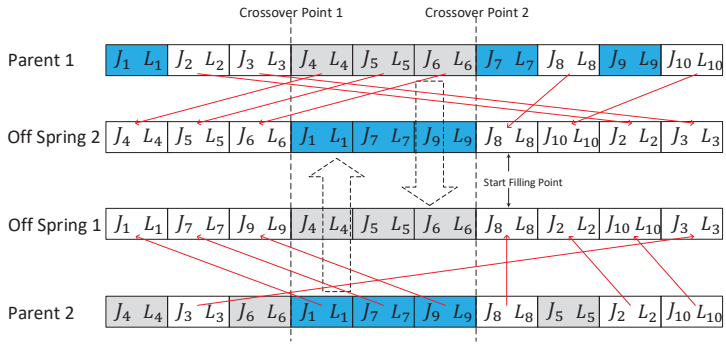


FIGURE 6 The crossover operation

5.6 | Iteration and Termination

The reproduction process as discussed above is repeated until a pre-defined maximum number of iterations is reached or the value of objective function of an individual is less than a pre-specified threshold. When the termination condition is reached, the fittest individual in the current population is the optimal solution. Note that the proposed system may fail to generate a viable solution, if too many jobs need to be scheduled which are beyond the maximum manufacturing capacity. In this case, the GA process will keep trying until the maximum number of iterations has been reached. This may be addressed by applying soft flexible scheduling techniques (Miguel and Shen (2003)), which searches a solution that minimises the cost of constraint violation. This extension is beyond the scope of the current project, which remains as a piece of future work.

6 | EXPERIMENTATION

The two main components of the proposed system are validated and evaluated int this section using illustrative examples and a simulated data set.

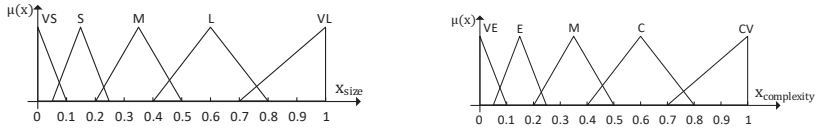
6.1 | Illustrative Examples on Rule Base Evolvement

Three illustrative examples are taken in this section to demonstrate the working procedure of the proposed completion time estimation system. In these examples, the value of λ in Equation 4 is pre-defined as 0.5, the threshold for similarity degree (δ) is set to 0.8, and the threshold (φ) for weight comparison is set to 0.8. For simplicity and to facilitate comprehensibility, only triangular membership functions are used in the example to represent fuzzy sets.

6.1.1 | Model Construction

The model takes 7 inputs, as listed in Table 1, which jointly describe a given manual collate and pack task, and it predicts the time of completion for a particular type of lane of certain size. In particular, two inputs, including the object size and the task complexity, take values from fuzzy partitioned variable domain based on the domain knowledge as shown

in Figure 7; and the other inputs are fuzzy numbers. These fuzzy numbers are usually provided by the in-house job estimating team. For instance, for a given task, the in-house job estimating team may estimate that the number of parts is between 5 and 8, but it is most likely to be 7. Then a triangular fuzzy number (4, 7, 8) will be used as the input for the attribute of the number of parts.



(a) Object size (b) Task complexity

FIGURE 7 Fuzzy partition of antecedent attributes

6.1.2 | Scenario 1

Suppose that the initialised rule base only contains 5 fuzzy rules (R_1^1, R_1^2, R_4^3, R_4^4, *and* R_8^5), which is extracted from the domain knowledge of the area manager, as shown in Table 4. There is a collate and pack task given as $I = (x_1 = (0.1, 0.2, 0.3), x_2 = (0.1, 0.2, 0.3), x_3 = (4, 5, 6), x_4 = (2.0, 2.2, 2.8), x_5 = (3, 4, 5), x_6 = (2, 4, 5), x_7 = (2, 3, 5))$, which obviously does not overlap with any rule antecedents in the given rule base. In order to enable global planning and scheduling by an intelligent job shop scheduling system, the completion times of the task on various lanes are estimated first using FRI (and the transformation-based FRI is particularly used in this example).

TABLE 4 The initialised rule base

No.	Antecedents							Consequents							
	x_1	x_2	x_3	x_4	x_5	x_6	x_7	k=1	k=2	k=3	k=4	k=5	k=6	k=7	k=8
1	(0.05,0.15,0.25)	(0.05,0.15,0.25)	(1,2,3)	(0.8,1.5,1.8)	(1,2,3)	(1,2,3)	(1,2,3)	7.2(1)							
2	(0.05,0.15,0.25)	(0.05,0.15,0.25)	(4,6,7)	(3.0,3.5,4.0)	(5,7,8)	(4,6,8)	(3,4,5)	12.0(1)							
3	(0.2,0.35,0.5)	(0.4,0.6,0.8)	(7,9,10)	(8.2,10.5,11.2)	(6,7,8)	(7,9,10)	(5,6,7)				4.9(1)				
4	(0.4,0.6,0.8)	(0.7,1,1)	(12,14,15)	(18.5,20.8,22.5)	(7,9,10)	(13,14,16)	(7,8,9)				8.9(1)				
5	(0.7,1,1)	(0.7,1,1)	(17,18,20)	(28.5,30.5,31.2)	(10,11,12)	(18,20,21)	(9,10,11)								6.0(1)

There are two rules available with the consequence being the completion time on a lane with 1 worker, two rules available with consequence being the completion time on a lane with 4 workers, and one rule available with consequence being the completion time on a lane with 8 workers in the initialised rule base. From these rules, the completion time required for the given task on a lane with 1 worker (i.e., $z_1 = 9.37$) can be interpolated from neighbouring rules R_1^1 and R_1^2; and that on a lane with 4 workers (i.e., $z_4 = 2.24$) can be extrapolated using neighbouring rules R_4^3 and R_4^4. From this, the time of completion on the rest of lanes for the given task can either be interpolated or extrapolated based on the previously generated rules R_1^* and R_4^* by artificially taking the number of workers on the lane as an input attribute. The generated results are listed in Table 5.

After feeding the results into the job shop scheduling system, assume that a globally optimised solution was generated which has planned the task on a lane with one worker, and such a lane was finally scheduled in the collate and pack area for the given task. Upon the completion of the task, the actual time taken for this task is $t = 9.85$. From this, the weight of the interpolated rule is calculated as $w_1^* = 0.95$ based on Equation 5. Therefore, the newly

TABLE 5 Times of completion for scenario 1

Antecedents							Consequents							
x_1	x_2	x_3	x_4	x_5	x_6	x_7	$k=1$	$k=2$	$k=3$	$k=4$	$k=5$	$k=6$	$k=7$	$k=8$
(0.1,0.2,0.3)	(0.1,0.2,0.3)	(4,5,6)	(2.0,2.2,2.8)	(3,4,5)	(2,4,5)	(2,3,5)	**9.37**	**4.69**	**3.12**	**2.24**	**1.87**	**1.56**	**1.34**	**1.17**

interpolated rule can be represented as:

$$
\begin{aligned}
R_1^*: \textbf{IF } & x_1 \text{ is } (0.1, 0.2, 0.3) \text{ and } x_2 \text{ is } (0.1, 0.2, 0.3) \text{ and} \\
& x_3 \text{ is } (4, 5, 6) \text{ and } x_4 \text{ is } (2.0, 2.2, 2.8) \text{ and} \\
& x_5 \text{ is } (3, 4, 5) \text{ and } x_6 \text{ is } (2, 4, 5) \text{ and} \\
& x_7 \text{ is } (2, 3, 5) \\
& \textbf{THEN } z_1^* = 9.37\ (0.95).
\end{aligned} \tag{11}
$$

According to the rule revision procedure discussed in Section 4.3, once a new rule has been interpolated, the similarity degree between the newly interpolated rule and the existing rules in the rule base are computed to determine the actions for rule base revision. Given the newly interpolated rule as shown above in Equation 11, the similarity degrees between it and all existing rules in the rule base are calculated. Note that there are only two rules R_1^1 and R_1^2 whose consequences are the time of completion on a lane with one worker. The sub-results and the final results of the calculation based on Equation 6 are summarised in Table 6. As the similarity degrees are less than the given threshold $\delta = 0.8$ and the weight of the rule ($w_1^* = 0.95$) is greater than the predefined threshold $\varphi = 0.8$, this interpolated rule R_1^* is added into rule base.

TABLE 6 The details of similarity degree calculation

i	$S(x_1^*, x_1^i)$	$S(x_2^*, x_2^i)$	$S(x_3^*, x_3^i)$	$S(x_4^*, x_4^i)$	$S(x_5^*, x_5^i)$	$S(x_6^*, x_6^i)$	$S(x_7^*, x_7^i)$	$S(z_1^*, z_1^i)$	$S(R_4^*, R_4^i)$
1	0.75	0.75	0.4	0.59	0.5	0.54	0.6	0.77	**0.61**
2	0.75	0.75	0.88	0.65	0.6	0.6	0.82	0.78	**0.73**

6.1.3 | Scenario 2

The rule base keeps being revised whilst the system performs inferences. Suppose that the rule base has now been updated as shown in Table 7, and a new task, which is similar to the one discussed in Scenario 1, appears. Therefore, the time lengths of completion on different lanes need to be estimated. Of course, the completion time on lanes with 1 worker and 4 workers can be directly acquired from rules R_1^2 and R_4^2. The time of completion on a lane with 2 workers can be interpolated either from R_2^1 and R_2^3, or R_1^2 and R_4^2 by artificially taking the number of workers as a rule antecedent. In particular, result $z_2^{2\prime} = 6.64$ is interpolated using neighbouring rules R_2^1 and R_2^3; and result $z_2^{2\prime\prime} = 6.99$ is interpolated using neighbouring rules R_1^2 and R_4^2 by considering the number of workers as an input attribute. In this case, the final result is estimated as the average of the two interpolated results based on Equation 4, which is $z_2^2 = 6.82$. The estimation of the completion time on other types of lanes can also be implemented in one of the ways discussed above, but the calculation details are omitted.

Suppose that the lane with 2 workers has been selected by the job scheduling system, and the actual time taken for this manual task is $t = 9.6$. Based on Equation 5, the weight of the interpolated rule is determined as $w_2^* = 0.79$. Once the newly interpolated rule is constructed, the similarity between this interpolated rule R_2^* and the existing

TABLE 7 The updated rule base

No.	Antecedents							Consequents							
	x_1	x_2	x_3	x_4	x_5	x_6	x_7	$k=1$	$k=2$	$k=3$	$k=4$	$k=5$	$k=6$	$k=7$	$k=8$
1	(0.05,0.15,0.25)	(0.05,0.15,0.25)	(1,2,3)	(0.8,1.5,1.8)	(1,2,3)	(1,2,3)	(1,2,3)	7.2(1)	**5.36(0.8)**		**1.7(0.8)**				
2	(0.1,0.2,0.3)	(0.1,0.2,0.3)	(4,5,6)	(2.0,2.2,2.8)	(3,4,5,)	(2,4,5)	(2,3,5)	**9.37(0.95)**			**2.24(0.98)**				
3	(0.05,0.15,0.25)	(0.05,0.15,0.25)	(4,6,7)	(3.0,3.5,4.0)	(5,7,8)	(4,6,8)	(3,4,5)	12.0(1)	**8.97(0.8)**		**2.9(0.8)**				
4	(0.2,0.35,0.5)	(0.4,0.6,0.8)	(7,9,10)	(8.2,10.5,11.2)	(6,7,8)	(7,9,10)	(5,6,7)				4.9(1)				
5	(0.4,0.6,0.8)	(0.7,1,1)	(12,14,15)	(18.5,20.8,22.5)	(7,9,10)	(13,14,16)	(7,8,9)				8.9(1)				
6	(0.7,1,1)	(0.7,1,1)	(17,18,20)	(28.5,30.5,31.2)	(10,11,12)	(18,20,21)	(9,10,11)								6.0(1)

rules R_2^1 and R_2^3 are calculated as: $S(R_2^*, R_2^1) = 0.59$ and $S(R_2^*, R_2^3) = 0.60$, which are both less than the given threshold $\delta = 0.8$. According to the rule base revision procedure as shown in Figure 3, the interpolated rule R_2^* therefore is ignored, and the rule base keeps unchanged.

6.1.4 | Scenario 3

Suppose that now another collate and pack task appears as $I = (x_1 = (0.5, 0.7, 0.9), x_2 = (0.4, 0.7, 0.8), x_3 = (10, 11, 12), x_4 = (19.5, 21.2, 22.0), x_5 = (7, 8, 9), x_6 = (11, 12, 13), x_7 = (7, 8, 9))$. The times of completion on lanes with 1 worker, 2 workers and 4 workers for the given task can be either interpolated or extrapolated from the rule base shown in Table 7. In particular, the completion time on lanes with 1 worker and 2 workers can be extrapolated from neighbouring rules (R_1^2, R_1^3) and (R_2^1, R_2^3), respectively; and the completion time on a lane with 4 worker can be interpolated from neighbouring rules R_4^4 and R_4^5. The time estimation based on other lane setup can then be interpolated or extrapolated from the previously interpolated and extrapolated rules by artificially taking the number of rules as an input attribute. The estimated results for all lane setups are listed in Table 8.

TABLE 8 Times of completion for scenario 3

Antecedents							Consequents							
x_1	x_2	x_3	x_4	x_5	x_6	x_7	$k=1$	$k=2$	$k=3$	$k=4$	$k=5$	$k=6$	$k=7$	$k=8$
(0.5,0.7,0.9)	(0.4, 0.7, 0.8)	(10, 11, 12)	(19.5, 21.2, 22.0)	(7,8 ,9)	(11,12,13)	(7,8,9)	**27.20**	**13.60**	**9.07**	**6.7**	**5.44**	**4.53**	**3.89**	**3.40**

The job scheduling system finally selects the lane with 4 workers during the operation optimisation stage. After the job is completed, the weight for this interpolated rule is calculated as $w_4^* = 0.78$. The degrees of similarity between this interpolated rule and the existing rules are computed as: $S(R_4^*, R_4^1) = 0.26$, $S(R_4^*, R_4^2) = 0.40$, $S(R_4^*, R_4^3) = 0.46$, $S(R_4^*, R_4^4) = 0.73$, $S(R_4^*, R_4^5) = 0.82$. It is clear that a similar rule R_4^5 to the newly interpolated rule exists in the rule base as $S(R_4^*, R_4^5) = 0.82$. Then, the weight of the interpolated rule R_4^* is compared with that of the existing rule R_4^5, and $w_4^5 = 1 > w_4^* = 0.78$. Consequently, the interpolated rule is discarded and the rule base remains as it was.

6.2 | Manual Assembly Job Planning and Scheduling

Due to the commercial sensitivity of real-world performance data, two experiments based on a simulation data set are reported in this section to demonstrate the working of the proposed job scheduling system and to confirm the power of the proposed system in improving manufacturing efficiency.

6.2.1 | Simulation 1 - System Demonstration

The data set contains 10 collating and packing jobs that need to be scheduled, which are with various complexities and different dispatch deadlines. The environment of the collate and pack area in this experiment is the same as the

environment introduced in Section 6.1, which is able to accommodate 8 different sizes of lanes. There are maximally 13 workers available per day (8 working hours) to operate multiple jobs on multiple lanes at the same time. Consequently, based on these requirements, a large number of lane combinations are available for lane planning and job scheduling. For instance, lanes with 1, 4 and 8 workers are able to be operated at the same time, and the combination of lanes with 1, 3, 4 and 5 workers are also allowed amongst others. The aim of this demonstration is to use the proposed GA-based JSS system to find an optimal solution to schedule the 10 jobs and plan their lanes, thus to minimise the temporal cost of collating and packing and also to provide the preview of the available manufacturing capacity for the marketing and sales team. Also, for operational efficiency and as a common practice, all jobs are better to be finished within one day, such that the work shop can be cleared, cleaned and prepared for the jobs on the following day. This has been used as a hard constraint in the experiments. In this experiment, the parameters of GA are configured as follow: crossover rate = 0.85, mutation rate = 0.05, maximum iteration = 10,000 and the termination condition for an optimal solution is the time cost is not getting any better in 200 iterations. These parameters are empirically determined in this experiment.

Assume that the completion times of each job on different lanes have been estimated by the proposed task completion time estimation system, which are listed in columns 2-9 and 12-19 in Table 9, and columns 10 and 20 indicate the dispatch deadline of the corresponding job in days. Based on the given deadlines of each job and the estimated completion time on each lane, the valid lanes to complete the individual jobs can be identified, as shown in Table 10. From this, the initialised population for the GA can be generated. Figure 8 shows two examples of randomly generated individuals. From the individual encoding method, the completion days for finishing all 10 jobs can be easily determined, which are 6 and 5 days, respectively, as also illustrated in the figure. Note that, during the population intialisation stage, the constraint of deadline is not considered for fast starting. Once the population has been initialised, the genetic operators of crossover and mutation, are employed to generate the next generation of population.

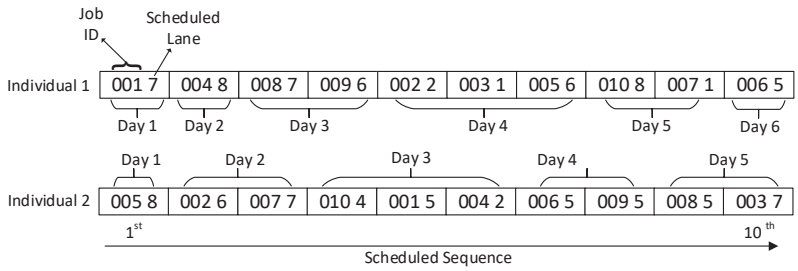


FIGURE 8 Two examples of randomly generated individuals for the initialised population

In this work, a two points crossover operation is adopted. Suppose that two parent individuals have been selected for crossover operation, as shown in Figure 8, and two crossover points, 3 and 6, have been randomly selected. The process of the crossover operation is illustrated in Figure 9, which reproduced two new scheduling solutions, named as offspring 1 and offspring 2. These two offspring individuals represent two new lane plans and job schedules for all the 10 jobs, which are shown in Table 11. In this situation, it is clear that a better solution has been obtained by offspring 1 after the crossover operation, which reduced the total completion days to 3 days from 6 days, at the

TABLE 9 Required completion times for the 10 simulated jobs

Job Number	Lane 1	Lane 2	Lane 3	Lane 4	Lane 5	Lane 6	Lane 7	Lane 8	Deadline (days)
1	12.80	12.43	10.28	8.66	7.05	5.55	4.57	2.99	2
2	8.97	7.11	5.93	5.78	4.14	3.15	1.98	0.99	2
3	7.70	6.72	5.74	4.75	3.70	2.79	1.81	0.82	4
4	9.25	7.98	7.02	5.91	4.63	3.68	2.57	1.45	2
5	16.21	14.75	12.92	10.35	10.47	7.66	6.18	4.07	2
6	14.98	13.48	11.77	10.17	7.98	6.96	5.37	3.76	3
7	7.74	6.76	5.77	4.78	3.54	2.81	1.83	0.84	4
8	19.02	17.36	15.43	13.36	11.23	9.35	7.44	5.38	4
9	16.64	15.77	13.30	10.80	10.77	7.79	6.46	4.35	2
10	13.00	11.65	9.13	7.70	6.94	5.83	4.41	2.96	2

TABLE 10 The smallest valid lane for each job

	Job 1	Job 2	Job 3	Job 4	Job 5	Job 6	Job 7	Job 8	Job 9	Job 10
Smallest valid lane	4	7	8	7	3	4	8	2	3	5

same time each job can be implemented in the specified deadline. However, offspring 2 leads to late completion for jobs 5 and 7, which consequently make an invalid solution. In this case, only offspring 1 will be used to replace often one of the worst individuals in the current population to produce the next generation of population. As an invalid solution, offspring 2 is ignored. The mutation operation in this work randomly mutates the scheduled lane number of a single job in a selected individual. In this demonstration, taking offspring 1 in Figure 9 as an example, the mutation point 9 has been randomly selected, and the scheduled lane size of job 6 in gene 9 is mutated to 8. As a result, the completion day of the new solution after the mutation operation was increased to 4 days from 3 days. Although it is not an optimal solution compared with current best individual, it is still a valid solution, as all jobs can be completed in time. Therefore, this offspring from mutation operation will still be used to replace one of the individuals usually with the worst fitness in the current population to generate a new generation of population. After a certain number of iterations, the GA is terminated; and the fittest individual in the population, as shown in Figure 10, indicates the optimal job schedule for the 10 current jobs. Of course, when a brand new job appears in the waiting list, the JSS system will be re-employed to update the lane plan and job scheduling. This will guarantee the planned lanes and scheduled jobs are always optimal and also the available capacity can be seen for a given deadline when the sales team commits to taking on a new job.

6.2.2 | Simulation 2 - System Evaluation

Currently, many small and medium-sized enterprises (SMEs) and some large manufacturers still manually implement planning and scheduling 'collation activities' outside the core MIS functionality, usually utilising tools, such as spreadsheets embedded with crude heuristics. Due to jobs' complexities, manual scheduling in this situation is often practically implemented as a simple first in and first out queue (FIFO) or an Earliest Deadline First queue (EDF) driven by

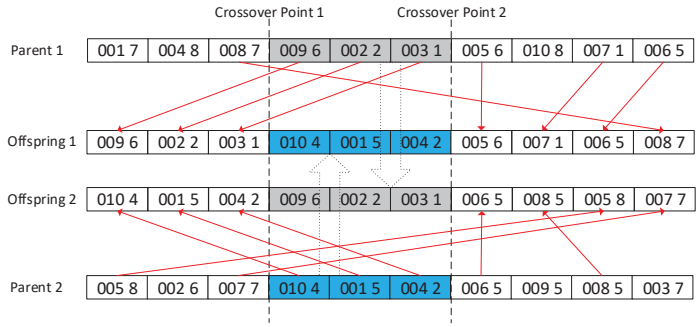


FIGURE 9 Crossover operation for two selected individuals

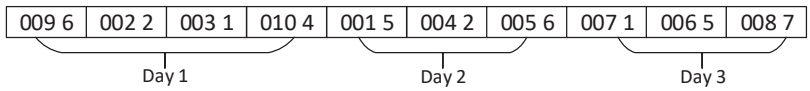


FIGURE 10 The optimal solution

dispatching deadlines. Briefly, FIFO is the simplest scheduling algorithm by which the jobs first taken by the manufacturer are assumed to be manufactured first. EDF is a dynamic scheduling algorithm to place manufacturing jobs in a priority queue based on the deadlines. Whenever a new job occurs, the queue will be updated and the jobs with the closest deadlines will be manufactured first. Consequently, the optimal scheduling solution of all such operations for all the current jobs is hard to achieve, which not only restricts the efficiency of manufacturing process, but also often leads to unnoticed late jobs until the times come close to the dispatch deadlines. In order to evaluate how the proposed scheduling system can improve the efficiency of the POS/POP manufacturing process, a simulated data set of 50 collating and packing jobs associated with deadlines is utilised in this section. The required completion times of each job on each lane can be estimated by the proposed task completion time estimation system, which are shown Table 12.

In order to estimate the proposed scheduling system in manufacturing efficiency, the proposed scheduling approach is compared with the ones usually used by the collate and pack manager. Note that the collate and pack jobs are only planned and scheduled by the area manager manually on three types of lanes with 1, 4 or 8 workers; and the jobs are scheduled on these three lanes using two commonly used planning methods which are the first come first served (FIFO) method and the earliest deadline first (EDF) method. Different to these manual approaches, the proposed scheduling system is able to efficiently consider all 8 lanes thanks to the computational and search ability of the algorithms, which minimises the time cost for the scheduling. The scheduling results using the three different methods are listed in Table 13. In addition, for comparison purposes, the proposed JSS system is also applied for job scheduling based only on three types of lanes, small, medium and large, to match the situations of manual panning methods, and the results are also reported in Table 13.

TABLE 11 Offspring 1 and Offspring 2 after the crossover operation

	Off Spring 1								Off Spring 2							
	Lane 1	Lane 2	Lane 3	Lane 4	Lane 5	Lane 6	Lane 7	Lane 8	Lane 1	Lane 2	Lane 3	Lane 4	Lane 5	Lane 6	Lane 7	Lane 8
Day 1	3	2		10		9				4		10	1			
Day 2		4			1	5			3	2				9		
Day 3	7				6		8						6		8	
Day 4																5
Day 5															7	

TABLE 12 The estimated time for completion on different sizes of lanes for the manual task data set

Arriving Order	Lane 1	Lane 2	Lane 3	Lane 4	Lane 5	Lane 6	Lane 7	Lane 8	Dispatch Deadline	Arriving Order	Lane 1	Lane 2	Lane 3	Lane 4	Lane 5	Lane 6	Lane 7	Lane 8	Dispatch Deadline
1	16.21	14.75	12.92	10.35	10.47	7.66	6.18	4.07	13	26	16.64	15.77	13.30	10.80	10.77	7.79	6.46	4.35	17
2	7.70	6.72	5.74	4.75	3.70	2.79	1.81	0.82	16	27	11.90	11.25	8.87	7.57	6.43	4.94	3.73	2.51	14
3	13.00	11.65	9.13	7.70	6.94	5.83	4.41	2.96	17	28	9.09	9.22	6.62	6.02	4.98	3.42	2.45	1.47	14
4	10.04	9.39	7.23	6.06	5.22	3.43	2.90	1.62	12	29	10.04	9.39	7.23	6.06	5.22	3.43	2.90	1.62	25
5	13.48	13.08	10.82	8.90	7.64	6.78	5.40	3.21	10	30	8.35	7.33	6.28	5.24	4.07	3.16	2.13	1.08	7
6	14.98	13.48	11.77	10.17	7.98	6.96	5.37	3.76	13	31	8.97	7.11	5.93	5.78	4.14	3.15	1.98	0.99	21
7	16.19	14.60	12.77	11.07	9.36	7.65	5.97	4.24	13	32	7.71	6.46	5.40	4.82	3.74	2.79	1.70	0.86	23
8	7.26	6.45	5.37	4.55	3.62	2.79	1.60	0.78	14	33	12.82	11.47	9.98	8.56	7.02	5.72	4.31	2.89	19
9	9.28	7.99	6.84	5.93	4.59	3.69	2.58	1.46	15	34	9.32	8.23	7.08	5.96	4.80	3.72	2.60	1.48	9
10	7.99	6.89	5.45	4.37	3.24	2.12	1.01	0.56	8	35	10.04	9.39	7.23	6.06	5.22	3.43	2.90	1.62	13
11	7.74	6.76	5.77	4.78	3.54	2.81	1.83	0.84	13	36	7.71	6.46	5.40	4.82	3.74	2.79	1.70	0.86	14
12	7.26	6.45	5.37	4.55	3.62	2.79	1.60	0.78	21	37	8.35	7.33	6.28	5.24	4.07	3.16	2.13	1.08	17
13	7.55	6.59	5.61	4.64	3.57	2.70	1.73	0.76	7	38	11.91	10.63	9.22	7.88	6.42	5.20	3.87	2.52	11
14	9.04	8.74	6.56	5.89	4.78	3.36	2.28	1.21	24	39	7.55	6.59	5.61	4.64	3.48	2.70	1.73	0.76	20
15	9.25	7.98	7.02	5.91	4.63	3.68	2.57	1.45	19	40	11.76	10.50	9.10	7.78	6.44	5.12	3.80	2.46	9
16	10.04	9.39	7.23	6.06	5.22	3.43	2.90	1.62	15	41	11.17	9.88	8.12	7.29	5.90	4.40	3.50	2.25	17
17	7.26	6.45	5.37	4.55	3.62	2.79	1.60	0.78	15	42	15.89	14.53	12.54	10.22	10.43	7.35	6.12	3.94	14
18	9.04	8.74	6.56	5.89	4.78	3.36	2.28	1.21	13	43	10.24	9.66	7.40	6.33	5.70	3.94	3.01	1.67	20
19	8.97	7.11	5.93	5.78	4.14	3.15	1.98	0.99	15	44	13.49	13.56	11.36	8.93	7.91	6.82	5.55	3.28	18
20	12.69	12.35	10.22	8.20	6.80	5.30	4.56	2.96	25	45	13.48	12.08	10.82	7.90	6.64	4.78	2.40	1.81	22
21	9.09	7.22	6.62	6.02	4.98	3.42	2.45	1.47	16	46	17.11	15.46	13.54	11.76	9.93	8.18	6.42	4.62	22
22	12.14	10.85	9.42	8.06	6.67	5.33	3.98	2.61	16	47	13.48	13.08	10.82	8.90	7.64	6.78	5.40	3.21	14
23	7.71	6.46	5.40	4.82	3.74	2.79	1.70	0.86	12	48	8.34	7.31	6.26	5.23	3.98	3.15	2.12	1.08	18
24	16.64	15.77	13.30	10.80	10.77	7.79	6.46	4.35	14	49	12.80	12.43	10.28	8.66	7.05	5.55	4.57	2.99	11
25	15.86	14.30	12.50	10.83	9.10	7.47	5.81	4.11	17	50	19.02	17.36	15.43	13.36	11.23	9.35	7.44	5.38	8

Table 13 clearly shows that all the 50 jobs can be efficiently scheduled within 14 days without any jobs delay or extra working hours by applying the proposed scheduling system, compared with FIFO method and EDF method that both require 17 days for completion. In particular, the FIFO method leads to multiple job delays, which can be very costly due to the breach of contracts or last minute premium shipping. Although the job schedule made by the EDF method does not cause any job delay, some of jobs require extra working hours to be finished on time, such as Jobs 50, 6 and 47, which consequently will increase the economic cost of the manufacturer. Interestingly, all 50 jobs can also be scheduled for completion using the proposed JSS system following the required dispatch deadlines without causing any extra hours, if only three packaging lanes are considered by the proposed system. However, in this case, the jobs progressing days will be increased to 20 days if this method is employed, which costs the business an additional 6 days.

TABLE 13 The job scheduling made by different methods ("′" indicates the miss of the dispatch deadline, and "*" represents the requirement of extra working hours)

	FIFO			EDF			The proposed System								The proposed System with three fixed lanes		
	Lane 1	Lane 4	Lane 8	Lane 1	Lane 4	Lane 8	Lane 1	Lane 2	Lane 3	Lane 4	Lane 5	Lane 6	Lane 7	Lane 8	Lane 1	Lane 4	Lane 8
Day 1	2	3	1	13	10	30		13		35			24		23	40	50
Day 2	5	4	6	50*	34	40	10		18	40	5				10	18	5
Day 3	8	9	7	49*	38	5	23		4	38	49				13	30	49
Day 4	10	11	12	23	4	1			34	30		1				4	38
Day 5	13	14	15	11	6*	7		11			6	7			11	34	1
Day 6	17	18	16	8	18	35	36		28	27	47				8	35	6
Day 7	19	21	20	28*	27	24	8	19	48				50		36	27	7
Day 8	23	22	24	36	47*	42		15			22	42				28	24
Day 9	25	27	26	17	9	16	2	17	21				46		17	48	42
Day 10	30′	28	29	2	19	21			29	41		26			2	16	26
Day 11	32	31	33	22*	3	25	39		16	14	20					21	22
Day 12	36	34′	35	37*	41	26	12	9	43				33		39	3	25
Day 13	39	37	38′	48*	15	44		31		45			25			37	10
Day 14	40′	41	42	39	43	33	32		37	3	44					9	13
Day 15	45	43	44	12	31	45									12	41	44
Day 16	47	48	46	32	14	46									32	15	33
Day 17		49′	50′		29	20										43	46
Day 18																45	31
Day 19																14	20
Day 20																29	

7 | CONCLUSIONS

This paper presented a manual job shopping planning and scheduling system using a GA algorithm and a dynamic fuzzy model. In particular, the fuzzy model, implemented by adapting the recently proposed experience-based fuzzy rule interpolation approach, estimates the time of completion for manual collate, assembly and pack tasks on different lanes with a variety of product configurations and sizes. This provides the necessary prerequisite in implementing the GA-based intelligent job shop scheduling systems which not only plans the lane arrangement but also schedules jobs based on the planned lanes. The system was validated and evaluated by illustrative examples and simulated experiments. The experimental results demonstrated the power of the proposed system in improving manufacturing productivity and efficiency. This is of high pragmatic and financial interest to the business as lower value work could be outsourced if it is found to cause delays to higher priority work.

Despite of the success, the work can be further improved in different directions. Firstly, the proposed system was developed particularly for POS and POP manufacturers in this paper, but the underpinning artificial intelligence approaches are applicable to any industry with manual tasks involved in the manufacturing processes. Secondly, the proposed system only considers a simple situation that every worker takes one job per day. This constraint can be released to further improve the manufacturing efficiency; the implementation of this requires further investigation. Thirdly, it is worthwhile to investigate how the constraints can be relaxed when too many jobs beyond the maximum manufacturing capacity need to be scheduled.

conflict of interest

Conflicts of interest: none

references

Ak, B. and Koc, E. (2012) A guide for genetic algorithm based on parallel machine scheduling and flexible job-shop scheduling. *Procedia - Social and Behavioral Sciences*, **62**, 817 – 823. World Conference on Business, Economics and Management (BEM-2012), May 4–6 2012, Antalya, Turkey.

Anand, E. and Panneerselvam, R. (2016) A study of crossover operators for genetic algorithm and proposal of a new crossover operator to solve open shop scheduling problem. *American Journal of Industrial and Business Management*, **6**, 774.

Baker, J. E. (1985) Adaptive selection methods for genetic algorithms. In *Proceedings of the 1st International Conference on Genetic Algorithms*, 101–111. Hillsdale, NJ, USA: L. Erlbaum Associates Inc.

Çaliş, B. and Bulkan, S. (2015) A research survey: review of ai solution strategies of job shop scheduling problem. *Journal of Intelligent Manufacturing*, **26**, 961–973.

Chaudhry, I. A. and Khan, A. A. (2016) A research survey: review of flexible job shop scheduling techniques. *International Transactions in Operational Research*, **23**, 551–591.

Chen, C., Parthaláin, N. M., Li, Y., Price, C., Quek, C. and Shen, Q. (2016) Rough-fuzzy rule interpolation. *Information Sciences*, **351**, 1 – 17.

Chen, S.-J. and Chen, S.-M. (2003) Fuzzy risk analysis based on similarity measures of generalized fuzzy numbers. *IEEE Transactions on Fuzzy Systems*, **11**, 45–56.

Chen, T., Shang, C., Su, P. and Shen, Q. (2018) Induction of accurate and interpretable fuzzy rules from preliminary crisp representation. *Knowledge-Based Systems*, **146**, 152 – 166. URL: http://www.sciencedirect.com/science/article/pii/S0950705118300546.

Cheng, S.-H., Chen, S.-M. and Chen, C.-L. (2016) Adaptive fuzzy interpolation based on ranking values of polygonal fuzzy sets and similarity measures between polygonal fuzzy sets. *Information Sciences*, **342**, 176 – 190.

Dawkins, R. (1982) *The Extended Phenotype*. Oxford Oxford University Press.

Ghédira, K. (2013) *Constraint satisfaction problems: csp formalisms and techniques*. John Wiley & Sons.

Gomes, M., Barbosa-Póvoa, A. and Novais, A. (2005) Optimal scheduling for flexible job shop operation. *International Journal of Production Research*, **43**, 2323–2353.

Huang, Z. and Shen, Q. (2008) Fuzzy interpolation and extrapolation: A practical approach. *Fuzzy Systems, IEEE Transactions on*, **16**, 13–28.

Johnson, L. A. and Montgomery, D. C. (1974) *Operations research in production planning, scheduling, and inventory control*, vol. 6. Wiley New York.

Kóczy, L. and Hirota, K. (1993) Approximate reasoning by linear rule interpolation and general approximation. *International Journal of Approximate Reasoning*, **9**, 197–225.

Li, J., Qu, Y., Shum, H. P. H. and Yang, L. (2017) *TSK Inference with Sparse Rule Bases*, 107–123. Cham: Springer International Publishing.

Li, J., Shum, H. P. H., Fu, X., Sexton, G. and Yang, L. (2016) Experience-based rule base generation and adaptation for fuzzy interpolation. In *2016 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE)*, 102–109.

Li, J., Yang, L., Qu, Y. and Sexton, G. (2018) An extended takagi–sugeno–kang inference system (tsk+) with fuzzy interpolation and its rule base generation. *Soft Computing*, **22**, 3155–3170.

Mamdani, E. H. (1977) Application of fuzzy logic to approximate reasoning using linguistic synthesis. *Computers, IEEE Transactions on*, **C-26**, 1182–1191.

Miguel, I. and Shen, Q. (2003) Fuzzy rrdfcsp and planning. *Artificial Intelligence*, **148**, 11 – 52.

Naik, N., Diao, R. and Shen, Q. (2017) Dynamic fuzzy rule interpolation and its application to intrusion detection. *IEEE Transactions on Fuzzy Systems*, **PP**, 1–1.

Pinedo, M. (2015) *Scheduling*. Springer.

Takagi, T. and Sugeno, M. (1985) Fuzzy identification of systems and its applications to modeling and control. *IEEE Transactions on Systems, Man, and Cybernetics*, **SMC-15**, 116–132.

Tan, Y., Li, J., Wonders, M., Chao, F., Shum, H. P. H. and Yang, L. (2016) Towards sparse rule base generation for fuzzy rule interpolation. In *2016 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE)*, 110–117.

Wang, Y. M., Yin, H. L. and Wang, J. (2009) Genetic algorithm with new encoding scheme for job shop scheduling. *The International Journal of Advanced Manufacturing Technology*, **44**, 977–984.

Yang, L., Chao, F. and Shen, Q. (2017a) Generalized adaptive fuzzy rule interpolation. *IEEE Transactions on Fuzzy Systems*, **25**, 839–853.

Yang, L. and Shen, Q. (2009) Towards adaptive interpolative reasoning. In *2009 IEEE International Conference on Fuzzy Systems*, 542–549.

Yang, L. and Shen, Q. (2011) Adaptive fuzzy interpolation. *Fuzzy Systems, IEEE Transactions on*, **19**, 1107–1126.

Yang, L. and Shen, Q. (2013) Closed form fuzzy interpolation. *Fuzzy Sets and Systems*, **225**, 1 – 22. Theme: Fuzzy Systems.

Yang, L., Zuo, Z., Chao, F. and Qu, Y. (2017b) Fuzzy interpolation systems and applications. In *Fuzzy Control Systems*. InTech.

Zadeh, L. (1965) Fuzzy sets. *Information and Control*, **8**, 338 – 353.